What Is Claimed Is:

1. An apparatus for verifying correctness of a behavioral model of a microcode machine, the microcode machine operable in a native state and an emulated state, the apparatus, comprising:

means for producing the native state;

means for producing the emulated state; and

means for comparing the native state and the emulated state.

2. The apparatus of claim 1, wherein the behavioral model comprises one of a hardware description language and a processor.

3. The apparatus of claim 1, further comprising:

means for determining a microcode entry point into a microcode storage device that stores microcode; and

means for generating microinstructions corresponding to the microcode, wherein a sequence of microinstructions corresponds to a macroinstruction, wherein the means for producing the native state includes means for executing the microinstructions, and wherein the means for producing the emulated state includes means for executing the macroinstruction.

4. A method for verifying the correctness of a processor behavioral model, wherein a processor operates in one of a native mode state and an emulated mode state, the method, comprising:

determining if a macroinstruction to be executed is a native instruction;

if the macroinstruction is a native instruction, executing the native instruction, the execution producing the native mode state of the processor;

if the macroinstruction is not a native instruction:

fetching the macroinstruction,

providing microinstructions corresponding to the macroinstruction, and

executing the microinstructions, the execution producing the native mode state of the processor;

executing the macroinstruction, the execution producing an emulated state of the processor; and

comparing the native mode state the of the processor with the emulated state of the processor.

5. The method of claim 4, further comprising checking the native mode state of the processor against the behavioral model.

6. The method of claim 5, wherein checking the native mode state of the processor against the behavioral model comprises checking for faults generated during execution of the native instruction.

7. The method of claim 4, wherein providing microinstructions comprises:

generating a microcode entry point, the microcode entry point indicating a location in a microcode storage where the microinstructions begin; and

expanding the microcode entry point.

8. The method of claim 7, wherein expanding the microcode entry point comprises:

reading the microcode storage;

generating alias fields for the microinstructions;

generating a control word; and

generating the microinstructions.

9. The method of claim 8, wherein generating the microinstructions comprises determining one or more of faults, traps, and exceptions.

10. The method of claim 4, further comprising:

testing the fetched macroinstruction for faults; and

if a fault is detected, generating a fault entry point into a microcode storage that stores microcode for use by the processor.

11. The method of claim 4, wherein executing the macroinstruction comprises executing the macroinstruction on an emulated mode reference simulator that verifies the microinstructions correctly emulates the macroinstruction.

12. A method for verifying the correctness of a behavioral model of a micro-coded machine, comprising:

executing a sequence of microinstructions on a native mode simulator, the execution producing a native mode state of the micro-coded machine, wherein the sequence of microinstructions corresponds to a macroinstruction;

executing the macroinstruction on an emulated mode reference simulator, the execution producing an emulated state of the micro-coded machine; and

checking the native mode state and the emulated state against the behavioral model.

13. The method of claim 12, further comprising:

determining the sequence of microinstructions, comprising:

reading the macroinstruction, and

determining an entry point into a microcode storage, the microcode storage storing microinstructions for execution by the micro-coded machine, wherein the entry point defines a first microinstruction in the microinstruction sequence; and

expanding the microcode entry point.

14. The method of claim 13, wherein expanding the microcode entry point comprises:

reading the microcode storage;

generating alias fields for microinstructions in the sequence of microinstructions;

generating a control word; and

generating the microinstructions.

15. The method of claim 14, wherein generating the microinstructions comprises determining one or more of faults, traps, and exceptions.

16. The method of claim 12, further comprising:

testing the macroinstruction for faults; and

if a fault is detected, generating a fault entry point into the microcode storage.

17. An apparatus that verifies the correctness of a processor behavioral model, comprising:

a microcode storage that stores microcode corresponding to microinstructions;

a microcode expander that reads the microcode storage;

a native mode reference simulator that executes microinstructions, wherein execution of the microinstructions produces a native mode state of a processor;

an emulated mode reference simulator that executes macroinstructions, wherein a macroinstruction comprises a sequence of microinstructions, and wherein execution of the macroinstruction produces an emulated mode state of the processor; and

a state checker that compares the native mode state and the emulated mode state to the behavioral model.

18. The apparatus of claim 17, further comprising an emulated instruction sequencer, wherein the native mode reference simulator reports faults and state or control changes to the emulated instruction sequencer, and wherein the emulated instruction sequencer determines a subsequent microinstruction for execution based on one or more of the faults and the state or control changes.

19. The apparatus of claim 17, wherein the processor behavioral model is written in hardware description language.

20. The apparatus of claim 17, wherein the emulated mode reference simulator is independent of the processor behavioral model.